Time Series Forecasting of Taxi Pickups

Table of Contents

Objectives

The main objective of this report is to use the yellow taxicab dataset to build a forecasting model that can predict the number of pickups for every/any hour of the day. This model will help the business to estimate the accurate number of passengers for every hour thereby allocating taxi fleets efficiently to meet these demands.

We will train different time series forecasting models to achieve this.

Brief description of the data set and a summary of its attributes

TLC Trip Record Data has 12 years (2009 –2020) worth of Data available but I have decided to analyze a subset of the 2015.

In 2015, passengers took nearly 300 million yellow cab rides in New York City. Working with the complete dataset for all these rides would require considerable time and computational resources. The 12 data files used represent two percent of the total trips sampled at random from each month.

I chose this data because it is useful for real-world applications such as:

- Predicting taxi duration for a trip
- Allocating taxi to zones/regions based on demand.
- Identify whether a toll fee will be paid

Below is the summary of the data attributes as described here

Attribute / Column	Description
VendorID	A code indicating the TPEP provider that provided the record. **1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.**
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
PULocationID	TLC Taxi Zone in which the taximeter was engaged.
DOLocationID	TLC Taxi Zone in which the taximeter was disengaged.
RateCodeID	The final rate code in effect at the end of the trip. **1= Standard rate** **2=JFK** **3=Newark** **4=Nassau or Westchester**

	5=Negotiated fare **6=Group ride**
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka "store and forward," because the vehicle did not have a connection to the server. **Y= store and forward trip** **N= not a store and forward trip**
Payment_type	A numeric code signifying how the passenger paid for the trip. **1= Credit card** **2= Cash** **3= No charge** **4= Dispute** **5= Unknown** **6= Voided trip**
Fare_amount	The time-and-distance fare calculated by the meter.
Extra	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
MTA_tax	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	Tip amount –This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.

Data Exploration

Joining the 12 files provided resulted in 2, 922, 266 instances in the dataset with no missing values. We have 12 numerical features, 4 categorical features, 2 datetime features and 1 integer. For easy visualization, I converted *VendorID, RateCodeID and payment_type* to their corresponding values.

Summary of some the numerical attributes shows that this data is not perfect and therefore needs some cleaning for example there is no way we can have a negative tip or zero passengers.

	Passenger_count	Trip_distance	Fare_amount	Tip_amount
min	0.0	0.0	-150.00	-2.7
max	9.0	14680110.0	410266.86	650

Data Cleaning and Feature engineering

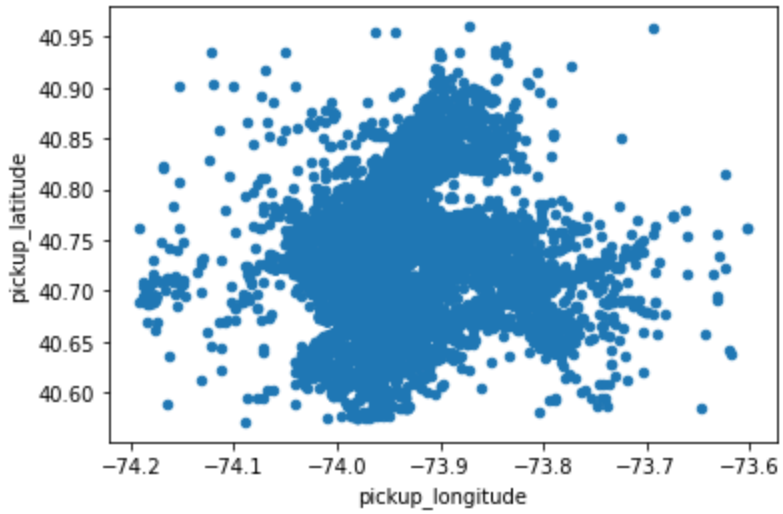
For data preprocessing/cleaning

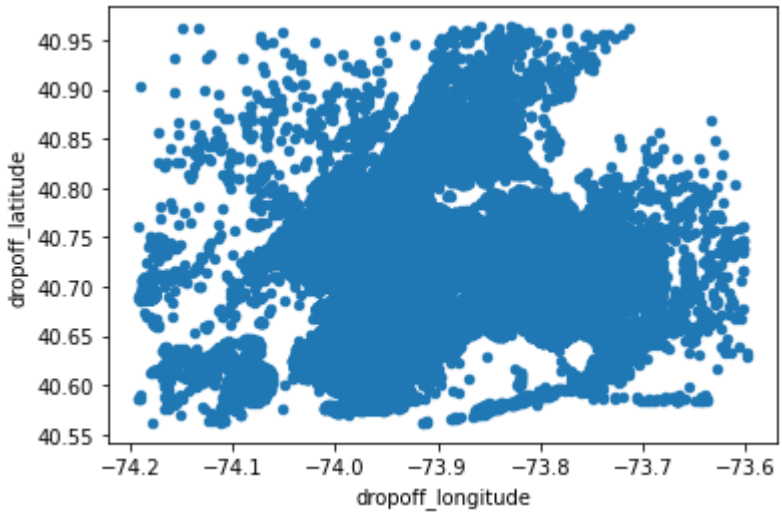
- Charges and trip information that are negative, as well as charges inconsistent with expected values are considered incorrect. In addition, trips with pickup or drop off locations outside a geographic region of interest are removed.
- Only keep trips with valid passenger and distance information.
- Remove trips missing valid pickup or drop off locations.
- Remove trips with invalid Rate code.

For Feature engineering, the following features were added.

- ***duration*** - Length of the trip, in minutes, calculated from the pickup and drop off times.
- ***avespeed*** - Average speed, in mph, calculated from the distance and duration values.
- ***time_of_day*** - This feature represents pick up time as the elapsed time since midnight in decimal hours (e.g. 7:10 am becomes 7.1667). The output is a duration vector with units of hours.
- ***day_of_week*** - This feature is a categorical array indicating the day of the week the trip began, in long format (e.g. 'Monday').
- ***toll_paid*** – This feature indicates trips that charged a toll or not.

After removing invalid trip information, we can now visualize some of the features.





Data Preparation for Machine Learning

Data Extraction

We extracted data for specific locations based on the below table

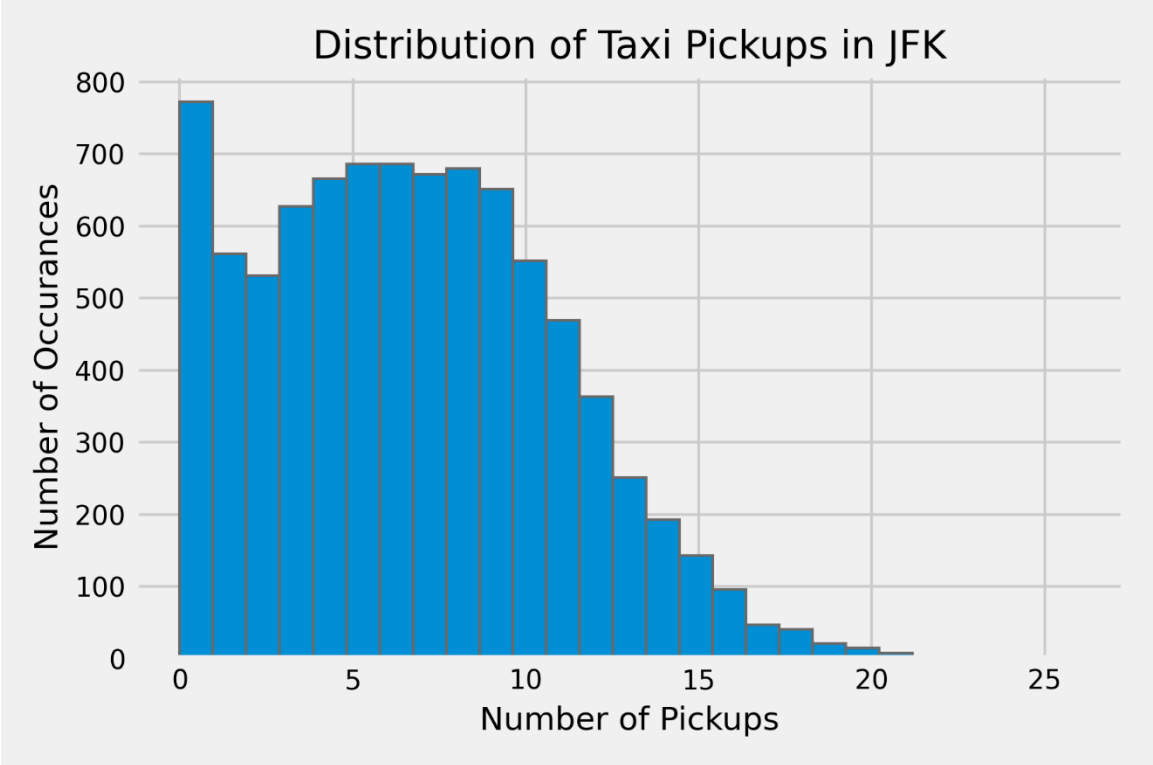
	Names	Lat1	Lat2	Lon1	Lon2
0	Manhattan	40.7485	40.7576	-73.9955	-73.9773
1	LaGuardia	40.7660	40.7760	-73.8760	-73.8610
2	JFK	40.6390	40.6500	-73.7930	-73.7750

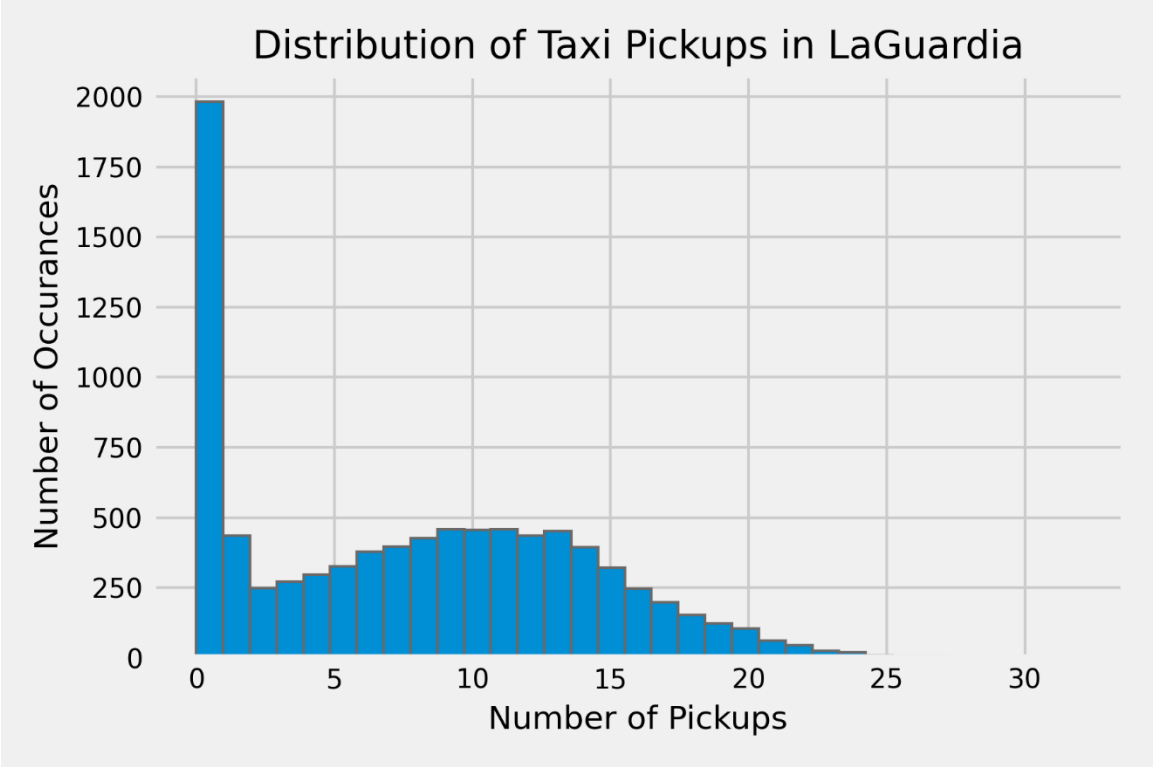
The table gives the latitude and longitude bounds for three pickup zones:

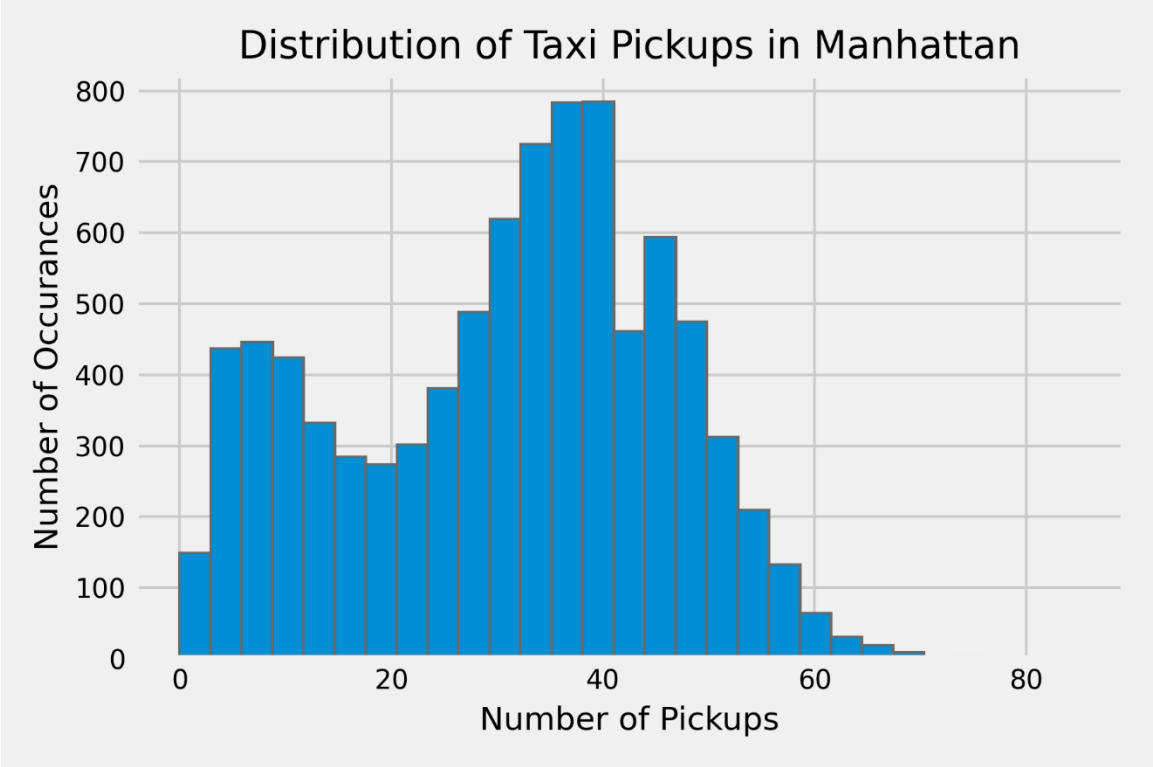
- **Manhattan**: here meaning an area of high taxi traffic surrounding Penn Station, Grand Central Station, and the Port Authority Bus Terminal
- **LaGuardia**: meaning an area surrounding LaGuardia airport
- **JFK**: similarly meaning an area surrounding JFK airport.

The extracted data represents the number of pickups in the original data over one-hour intervals of 2015 in the zones defined above. The sample data and distribution for each location is as shown below:

	PickupTime	Location	TripCount
0	2015-01-01 00:00:00	Manhattan	22
1	2015-01-01 00:00:00	LaGuardia	2
2	2015-01-01 00:00:00	JFK	2
3	2015-01-01 01:00:00	Manhattan	10
4	2015-01-01 01:00:00	LaGuardia	0



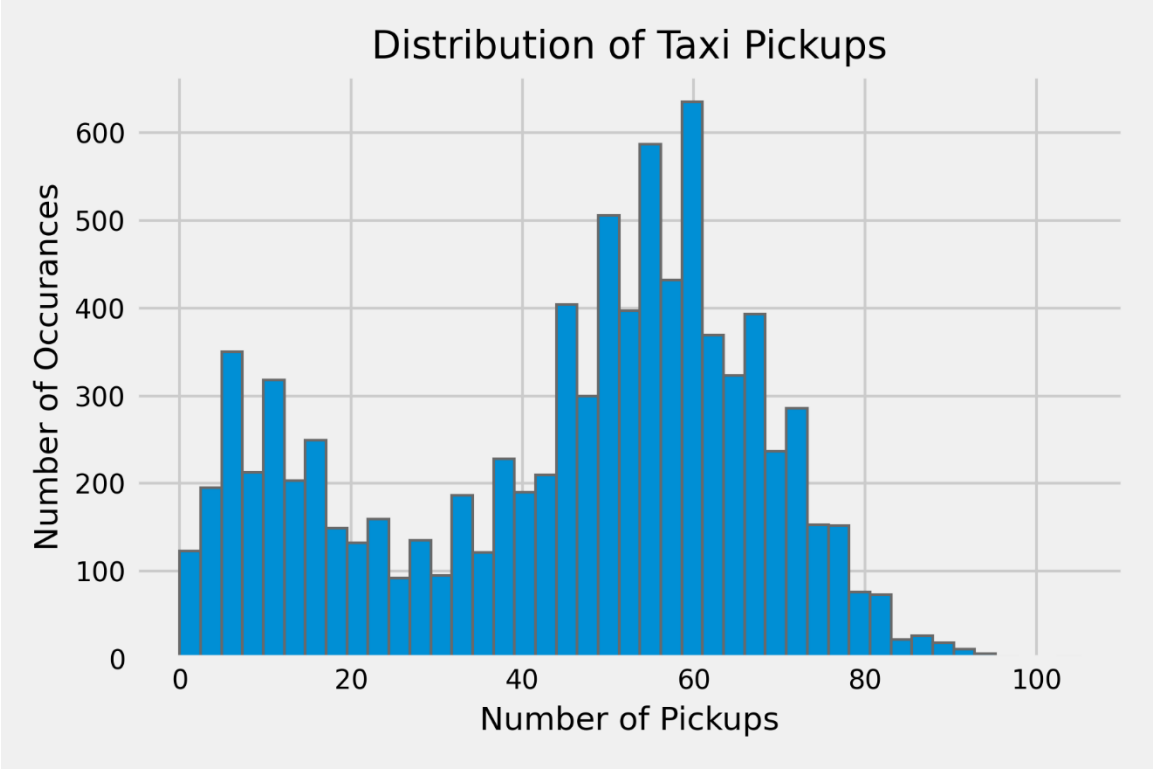

Distribution of Taxi Pickups in LaGuardia
Number of Occurances
2000
1750
1500
1250
1000
750
500
250
0
0
5
10
15
20
25
30
Number of Pickups


Distribution of Taxi Pickups in Manhattan
Number of Occurances
800
700
600
500
400
300
200
100
0
0
20
40
60
80
Number of Pickups

Data Merging

Finally, we merged the extracted data to show trip counts for all locations for every hour.

PickupTime	TripCount
2015-01-01 00:00:00	26
2015-01-01 01:00:00	12
2015-01-01 02:00:00	14
2015-01-01 03:00:00	9
2015-01-01 04:00:00	11



Train/Test Splitting

The data consist of 8760 hours of data, and we used 8700 hours for training and the remaining 60 hours to measure how accurate the trained model can predict taxi pickups for any given hour.

Model Training and Evaluation

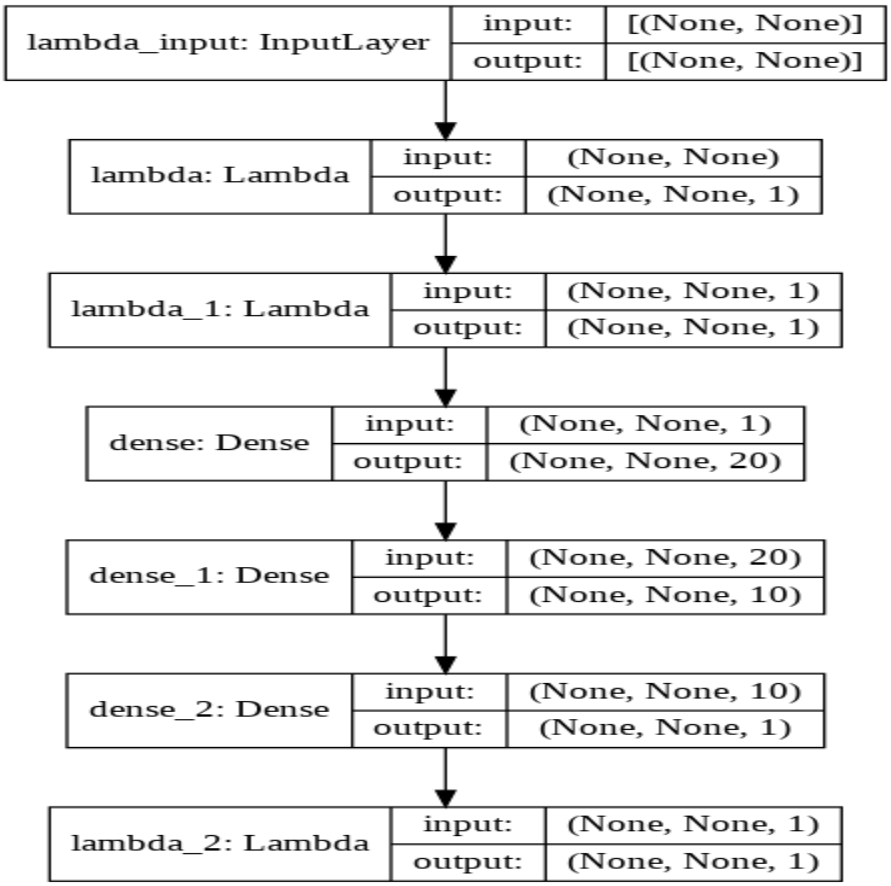
We trained four (4) different time series forecasting models which includes 1 Prophet model and 3 deep learning keras models with different architectures.

The configuration for the deep learning models:

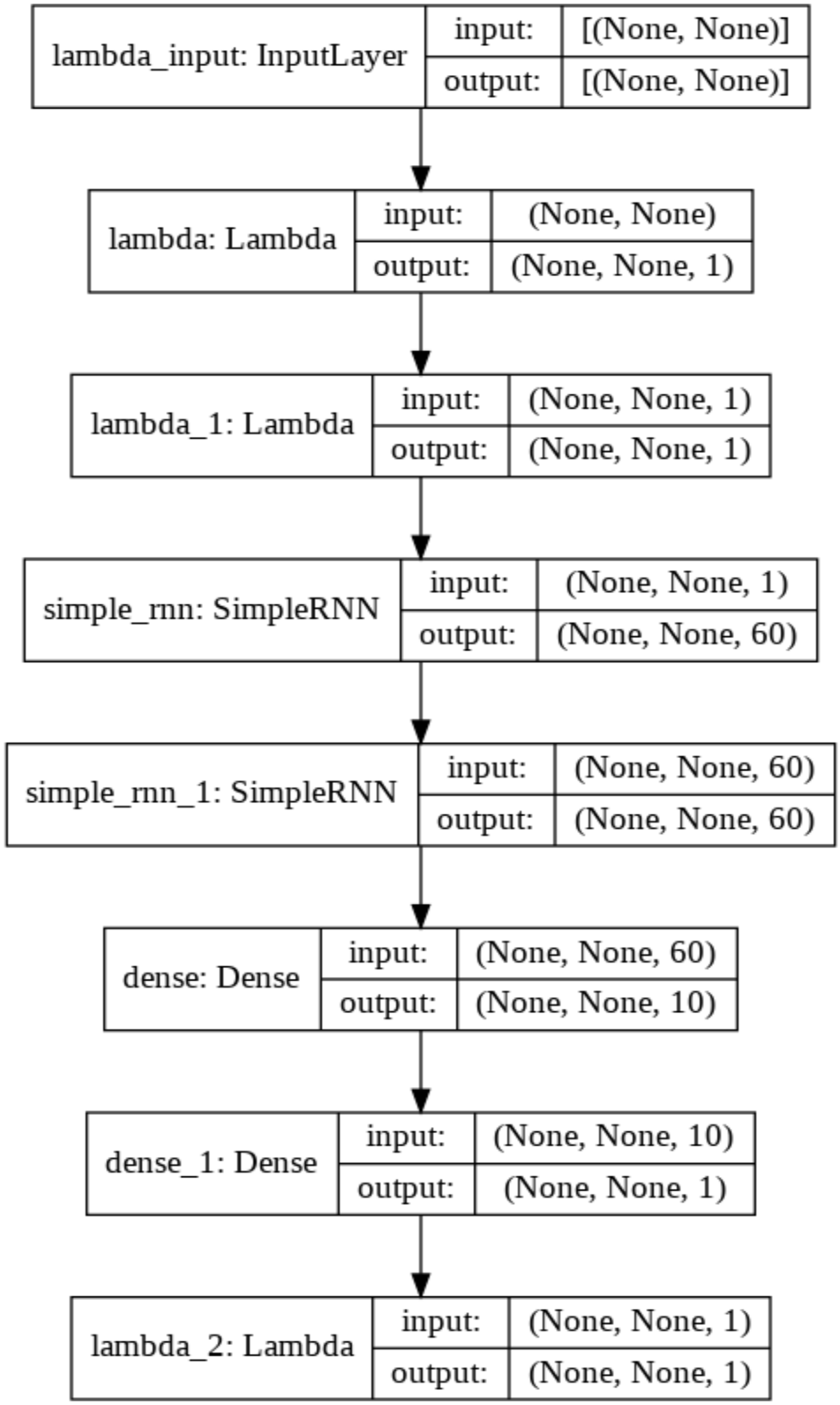
1. Train size: 8700 hours
2. Test size: 60 hours
3. Window size: 72
4. Batch size: 128
5. Epochs: 20
6. Loss function: Mean Squared Error (mse)
7. Optimizer: Adam with learning rate of 1e-3
8. Metrics: Mean Absolute Error (mae)

The architecture of the models are shown below

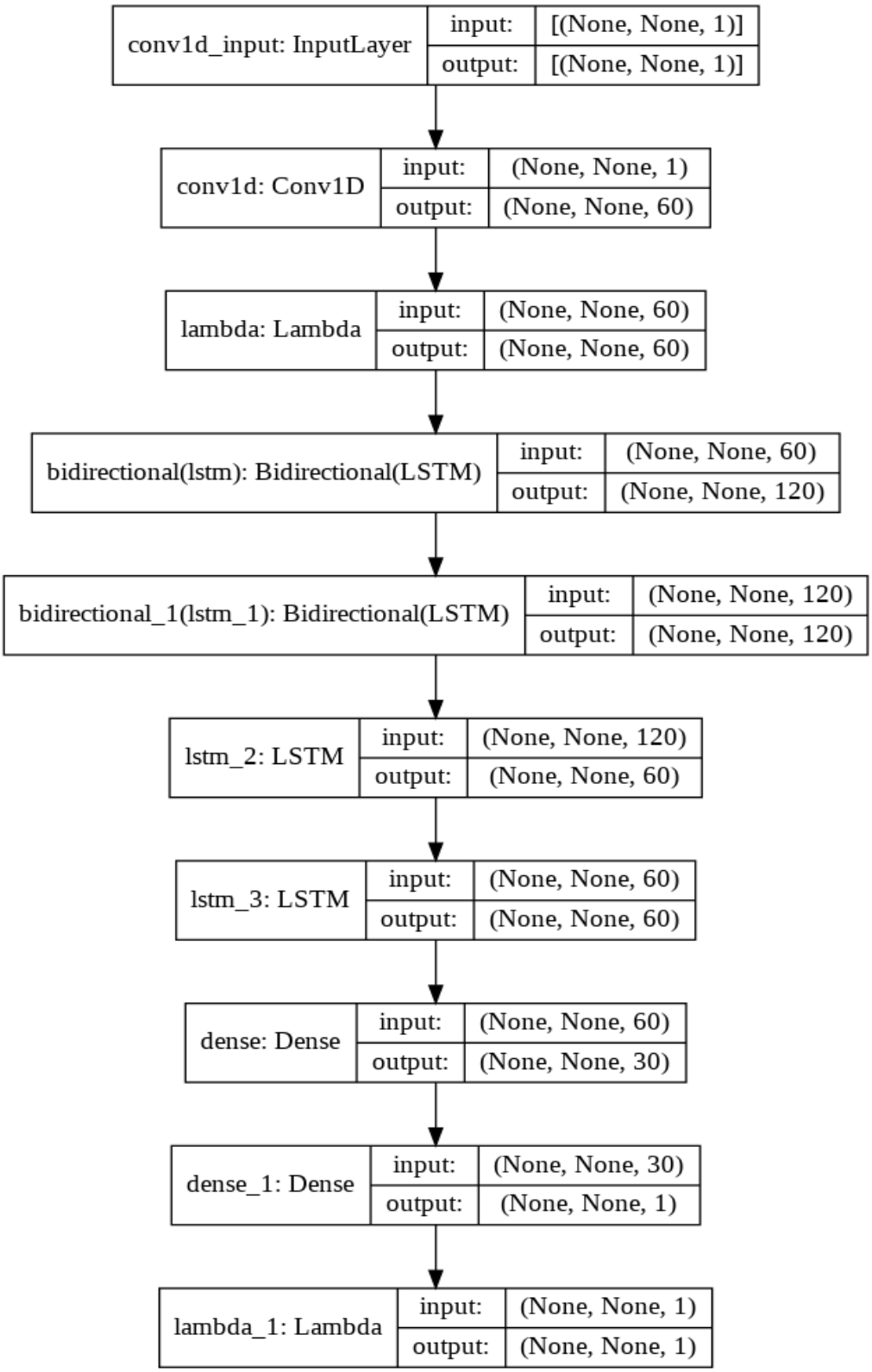
- Fully Connected



- Simple RNN


lambda_input: InputLayer
input: [(None, None)]
output: [(None, None)]
lambda: Lambda
input: (None, None)
output: (None, None, 1)
lambda_1: Lambda
input: (None, None, 1)
output: (None, None, 1)
simple_rnn: SimpleRNN
input: (None, None, 1)
output: (None, None, 60)
simple_rnn_1: SimpleRNN
input: (None, None, 60)
output: (None, None, 60)
dense: Dense
input: (None, None, 60)
output: (None, None, 10)
dense_1: Dense
input: (None, None, 10)
output: (None, None, 1)
lambda_2: Lambda
input: (None, None, 1)
output: (None, None, 1)

- Conv1D + Bidirectional LSTM + Fully Connected



We evaluated the models using Mean Absolute Error metric as shown below

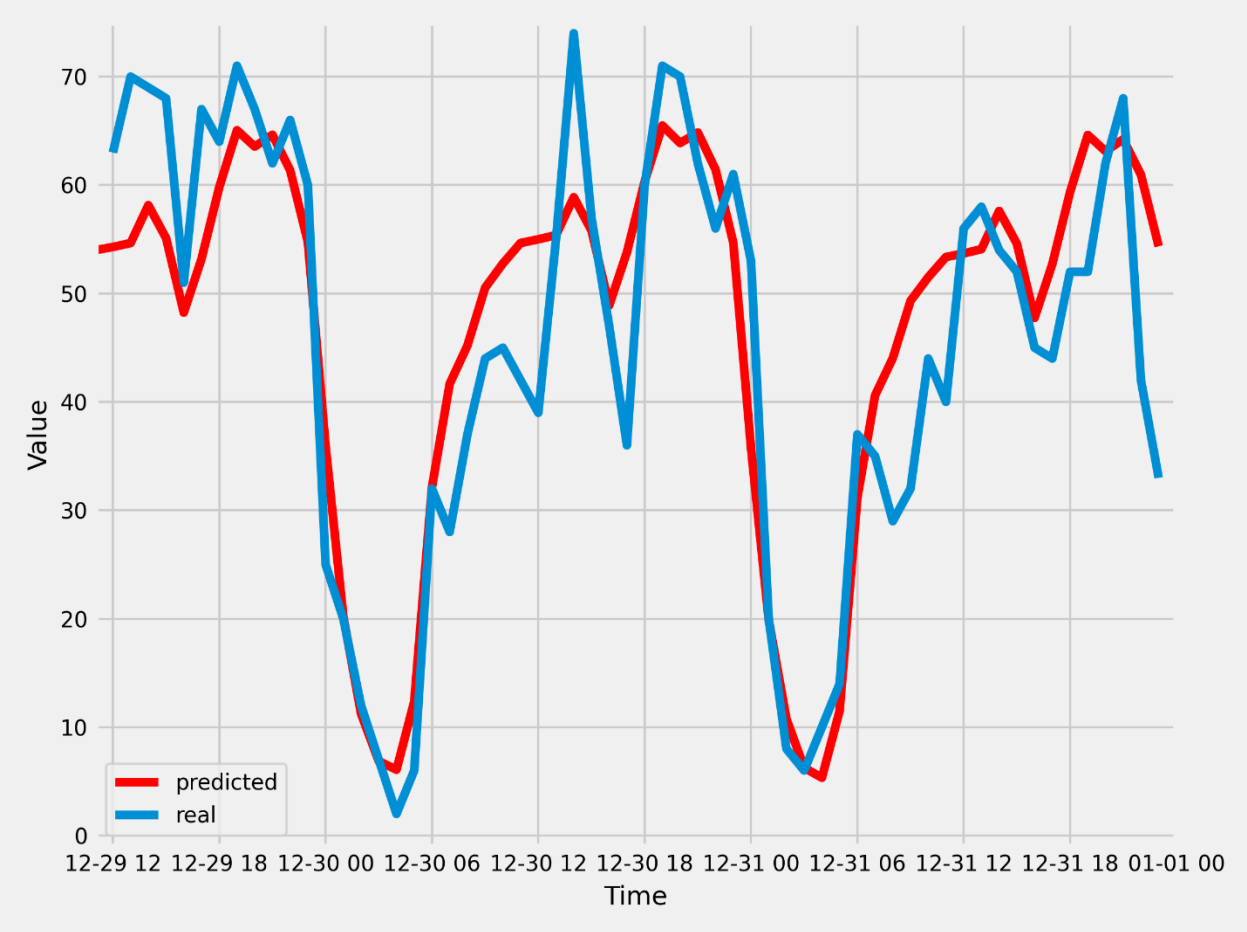
Model	Mean Absolute Error (mae)
Facebook Prophet	**7.035844**
Fully Connected	8.477009
Simple RNN	7.108506
Conv1D + Bidirectional LSTM + Fully Connected	7.542087

Key Findings and Insights

Using the table above, we can see that the facebook prophet model performed better than the deep learning models since it has the lowest mean absolute error.

This is vey surprising because deep learning models such as LSTM did not perform the best and this simply shows that deep learning model may not be the best for every situation.

The prediction for the model is as shown below:



Model Selection

The facebook prophet model seems to perform better and therefore chosen as the best model for our present case.

Next Steps in model improvement

As a next step, we will explore forecasting taxi pickups based on location as this will enable the business to allocate taxis efficiently for each location.

Summary

The selected model will be very useful in efficient allocation of taxis for every hour of the day. This will help the business to avoid waste thereby increasing revenue.